CLAIM AMENDMENTS

1. (Original) A method of performing software performance analysis for a target machine, comprising:

describing a system design as a network of logical entities;

selecting at least one of the logical entities for a software implementation;

synthesizing a software program from the logical entities selected for the software implementation;

compiling the software program to generate an optimized assembler code representation of the software program;

performing a performance analysis using the assembler code;

generating a software simulation model using the assembler code; and

generating a hardware/software co-simulation model using the software simulation model.

2. (Original) The method of claim 1, wherein the compiling step further comprises incorporating a description of the target machine.

3. (Original) The method of claim 1, wherein the software simulation model is an assembler-level C code simulation model.

4. (Original) The method of claim 1, further comprising selecting at least one of the logical entities for a hardware implementation, and synthesizing a software model of the hardware implementation from the selected logical entities, wherein the hardware/software co-simulation model is generated using the software model of the hardware implementation.

5. (Original) The method of claim 1, wherein the performance analysis measures an execution time of an element of the assembler code.

6. (Original) The method of claim 1, wherein the software program is compiled using the same compiler used to compile a production executable.

7. (Original) The method of claim 1, wherein performing the performance analysis comprises annotating the assembler code with performance information.

8. (Original) The method of claim 7, wherein the performance information is timing information.

9. (Previously Presented) A method of preparing software for a performance estimation, comprising:

providing a software assembly code module;

translating the assembly code module into a simulation model; and

annotating the simulation model with performance information.

10. (Previously Presented) The method of claim 9, wherein providing a software assembly code module comprises compiling software source code to assembly.

11. (Previously Presented) The method of claim 10, wherein the software assembly code module is compiled using a compiler adapted to create code that will execute on a first machine architecture.

12. (Previously Presented) The method of claim 11, wherein the performance information is associated with the first machine architecture.

13. (Previously Presented) The method of claim 11, wherein the simulation model is compiled to execute on a second machine architecture, the second machine architecture being different from the first machine architecture.

14. (Previously Presented) The method of claim 9, wherein providing a software assembly code module comprises disassembling software binary code to assembly code.

15. (Previously Presented) The method of claim 9, wherein the simulation model is an assembler-level representation of the software, expressed in a high-level programming language.

16. (Previously Presented) The method of claim 9, wherein the translation step further comprises gathering information from another software module.

17. (Previously Presented) The method of claim 16, wherein the information gathered comprises high-level hints about the software assembly code module.

18. (Previously Presented) The method of claim 9, wherein the performance information comprises estimated performance information.

19. (Previously Presented) The method of claim 9, wherein the performance information is statically estimated.

20. (Previously Presented) The method of claim 9, wherein the performance information is dynamically computed at run-time, using a formula provided during the annotating step.

21. (Previously Presented) The method of claim 9, further comprising:

compiling the simulation model to a simulator host program; and

executing the simulator host program on a simulator to allow performance measurements to be taken.

22. (Previously Presented) The method of claim 21, further comprising linking an already-annotated module with the simulation model.

23-32. (Cancelled).

33. (Previously Presented) A computer program product that includes a medium useable by a processor, the medium comprising a sequence of instructions which, when executed by said processor, causes said processor to execute a method for performing software performance analysis for a target machine, comprising:

describing a system design as a network of logical entities;

selecting at least one of the logical entities for a software implementation;

synthesizing a software program from the logical entities selected for the software implementation;

compiling the software program to generate an optimized assembler code representation of the software program;

performing a performance analysis using the assembler code;

generating a software simulation model using the assembler code; and

generating a hardware/software co-simulation model using the software simulation model.

34. (Previously Presented) The computer program product of claim 33, wherein the compiling step further comprises incorporating a description of the target machine.

35. (Previously Presented) The computer program product of claim 33, wherein the software simulation model is an assembler-level C code simulation model.

36. (Previously Presented) The computer program product of claim 33, further comprising selecting at least one of the logical entities for a hardware implementation, and synthesizing a software model of the hardware implementation from the selected logical entities, wherein the hardware/software co-simulation model is generated using the software model of the hardware implementation.

37. (Previously Presented) The computer program product of claim 33, wherein the performance analysis measures an execution time of an element of the assembler code.

38. (Previously Presented) The computer program product of claim 33, wherein the software program is compiled using the same compiler used to compile a production executable.

39. (Previously Presented) The computer program product of claim 33, wherein performing the performance analysis comprises annotating the assembler code with performance information.

40. (Previously Presented) The computer program product of claim 39, wherein the performance information is timing information.

41. (Previously Presented) A computer program product that includes a medium useable by a processor, the medium comprising a sequence of instructions which, when

executed by said processor, causes said processor to execute a method for preparing software for a performance estimation, comprising:

providing a software assembly code module;

translating the assembly code module into a simulation model; and

annotating the simulation model with performance information.

42. (Previously Presented) The computer program product of claim 41, wherein providing a software assembly code module comprises compiling software source code to assembly.

43. (Previously Presented) The computer program product of claim 42, wherein the software assembly code module is compiled using a compiler adapted to create code that will execute on a first machine architecture.

44. (Previously Presented) The computer program product of claim 43, wherein the performance information is associated with the first machine architecture.

45. (Previously Presented) The computer program product of claim 43, wherein the simulation model is compiled to execute on a second machine architecture, the second machine architecture being different from the first machine architecture.

46. (Previously Presented) The computer program product of claim 41, wherein providing a software assembly code module comprises disassembling software binary code to assembly code.

47. (Previously Presented) The computer program product of claim 41, wherein the simulation model is an assembler level representation of the software, expressed in a high-level programming language.

48. (Previously Presented) The computer program product of claim 41, wherein the translation step further comprises gathering information from another software module.

49. (Previously Presented) The computer program product of claim 48, wherein the information gathered comprises high-level hints about the software assembly code module.

50. (Previously Presented) The computer program product of claim 41, wherein the performance information comprises estimated performance information.

51. (Previously Presented) The computer program product of claim 41, wherein the performance information is statically estimated.

52. (Previously Presented) The computer program product of claim 41, wherein the performance information is dynamically computed at run-time, using a formula provided during the annotating step.

53. (Previously Presented) The computer program product of claim 41, further comprising:

compiling the simulation model to a simulator host program; and

executing the simulator host program on a simulator to allow performance measurements to be taken.

54. (Previously Presented) The computer program product of claim 53, further comprising linking an already-annotated module with the simulation model.

55. (Previously Presented) A method of translating an assembly language software module into a simulation model, comprising:

receiving the assembly language software module,

parsing the assembly language software module into a data structure, the data structure comprising one or more nodes, each of the one or more nodes being mapped to a period of time using a mapping definition, each of the one or more nodes containing an element of the assembly language software module;

processing the data structure to refine the accuracy of the simulation model;

associating performance information with an element of the assembly language software module; and

outputting the simulation model.

56. (Previously Presented) The computer program product of claim 55, wherein the one or more nodes comprises a first node and a second node, the first node being mapped to a

first period of time, the second node being mapped to a second period of time, the first period of time being different from the second period of time.

57. (Previously Presented) The computer program product of claim 55, wherein the performance information comprises an execution delay value for the element of the assembly language software module.

58. (Previously Presented) The computer program product of claim 55, wherein the performance information is a statically computed value.

59. (Previously Presented) The computer program product of claim 55, wherein the performance information is a formula for dynamically computing a value.

60. (Previously Presented) The computer program product of claim 55, wherein processing the data structure comprises replicating the behavior of the assembly language software model in the simulation model.